

**A Lecture Series on
DATA COMPRESSION
JPEG and MPEG Standards**

Abdou Youssef (Speaker)

Anastase Nakassis (MDVTG Manager)

Motivation for Standards

- Why Standards?
 - Compatibility
 - Production cost reduction
 - Triggering growth and product development
- Do JPEG/MPEG standards kill research?
 - Not necessarily
 - Those standards are very flexible regarding the encoder design, leaving much room for improvement
 - The trends are toward even greater flexibility in future generations of the standards
 - Another growth area is the development of systems which integrate components that use the standards

Image/Video Compression Standards (Outline)

- JPEG
 - Baseline JPEG
 - Extended JPEG
 - Lossless JPEG (DPCM + Huffman/Arithmetic)
- MPEG
 - History (H.261 and H.263)
 - MPEG1
 - MPEG2
 - Why not MPEG3
 - Future: MPEG4 (slated for 1998)

The JPEG “Toolkit”

- JPEG provides a “toolkit” of techniques for compressing continuous-tone, still, color and monochrome images
- Baseline JPEG provides a DCT-based algorithm, and uses run-length encoding and Huffman coding
- Baseline JPEG operates only in sequential mode, and is restricted to 8 bits/pixel input images
- Extended JPEG offers several optional enhancements:
 - 12-bit/pixel input
 - Progressive transmission
 - Choice between Arithmetic and Huffman coding
 - Adaptive quantization
 - Tiling
 - Still picture interchange file format (SPIFF)
 - Selective refinement
- Applications of JPEG
 - Desktop publishing, color fax, photojournalism, picture archiving and communications systems for medical images, general image archiving systems, consumer imaging, graphic arts, and others

The Baseline JPEG Algorithm

1. It operates on 8×8 blocks of the input image
2. Mean-normalization (subtract 128 from each pixel)
3. Transform: DCT-transform each block
4. Quantization
 - An 8×8 quantization matrix Q is user-provided
 - Each block is divided by Q (point by point)
 - The terms are then rounded to their nearest integers
 - Remark: Up to 4 quantization matrices per image are allowed (for example, one for luminance, and for each of the three color components)
5. Entropy-coding of the DC coefficients (the top left coefficient of each quantized block) using DPCM+Huffman
 - Huffman-encode the DC residuals derived from the difference between each DC and the DC of the preceding block
6. Entropy-coding of the AC (i.e., non-DC) coefficients
 - Zigzag-order the quantized coefficients of each block
 - Record for each nonzero coefficient both its distance (called *run*) to the preceding nonzero coefficient in the zigzag sequence, and its value (called *level*)
 - Huffman code the $[\text{run}, \text{level}]$ terms using one single Huffman table for all the AC's of the image

A Schematic Diagram of the JPEG Encoder

The Quantization Matrices

- They are user-provided
- They can be computed using the contrast sensitivity function of the HVS
- Their values are 8-bit integers
- They provide control over the bitrate by scaling them by a constant factor
- Example

$$Q = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}.$$

The DC Huffman Table

- The DC residuals are in the range $[-2047, 2047]$
- Divide this range into 12 categories corresponding to “one-dimensional ripples”
- Category k , $0 \leq k \leq 11$, consists of 2^k integers ranging from $-2^k + 1$ to -2^{k-1} , and from 2^{k-1} to $2^k - 1$
- Develop a Huffman code for the 12 categories, where every codeword is at most 16 bits long
- Encode each DC residual as a binary string hsm where
 - h is the codeword of the residual’s category
 - s = sign of the residual; $s = 0$ if negative, 1 if positive
 - m = the $(k - 1)$ -bit $[(\text{magnitude of the residual}) - 2^{k-1}]$

Coding of the AC Terms (The AC Huffman Table)

- Divide the range of the AC's into 10 categories
- Category k , $1 \leq k \leq 10$, consists of 2^k integers ranging from $-2^k + 1$ to -2^{k-1} , and from 2^{k-1} to $2^k - 1$
- Describe each $[run, level]$ by 8 bits 'NNNNSSSS'
 - NNNN = the value of run (i.e., the runlength)
 - SSSS = the category of $level$
 - If the runlength $run > 15$, then
 - * $run = 15p + r$, $0 \leq r \leq 15$, $r = r_3r_2r_1r_0$ in binary
 - * describe $[run, level]$ by
 $11110000_1 \ 11110000_2 \ \dots \ 11110000_p \ r_3r_2r_1r_0SSSS$
- Add an end-of-block *EOB* symbol after the last nonzero coefficient in each block
- The total number of symbols needed is $16 \cdot 10 + 1 + 1 = 162$
- Build a Huffman table for those 162 symbols, where every codeword is at most 16 bits long
- Encode each AC quantized term as hsm where
 - h is the codeword of the AC's $[run, category]$
 - s = sign of the term; $s = 0$ if negative, 1 if positive
 - m = the $(k - 1)$ -bit $[(\text{magnitude of the term}) - 2^{k-1}]$

Examples

- Take an 8×8 block of Lena

$$B = \begin{pmatrix} 143 & 147 & 149 & 152 & 156 & 147 & 146 & 149 \\ 151 & 146 & 143 & 154 & 148 & 144 & 153 & 132 \\ 147 & 143 & 145 & 149 & 144 & 145 & 128 & 133 \\ 152 & 145 & 145 & 144 & 146 & 134 & 130 & 137 \\ 146 & 143 & 142 & 147 & 124 & 127 & 139 & 138 \\ 139 & 145 & 139 & 127 & 126 & 135 & 139 & 141 \\ 145 & 137 & 124 & 130 & 138 & 136 & 140 & 144 \\ 144 & 124 & 136 & 134 & 137 & 139 & 142 & 145 \end{pmatrix}$$

- Normalize B

$$NB = \begin{pmatrix} 15 & 19 & 21 & 24 & 28 & 19 & 18 & 21 \\ 23 & 18 & 15 & 26 & 20 & 16 & 25 & 4 \\ 19 & 15 & 17 & 21 & 16 & 17 & 0 & 5 \\ 24 & 17 & 17 & 16 & 18 & 6 & 2 & 9 \\ 18 & 15 & 14 & 19 & -4 & -1 & 11 & 10 \\ 11 & 17 & 11 & -1 & -2 & 7 & 11 & 13 \\ 17 & 9 & -4 & 2 & 10 & 8 & 12 & 16 \\ 16 & -4 & 8 & 6 & 9 & 11 & 14 & 17 \end{pmatrix}$$

- Perform DCT

$$D = \begin{pmatrix} 103.4 & 12.4 & 6.0 & 2.1 & 8.1 & 5.7 & -0.7 & -0.4 \\ 31.7 & 11.6 & -22.8 & -0.7 & -0.1 & -0.5 & -4.7 & -1.8 \\ 11.0 & -21.2 & -3.7 & 8.8 & 2.3 & 0.1 & -0.2 & 5.6 \\ 0.2 & -4.9 & 10.3 & -8.6 & -8.9 & -2.5 & -7.6 & -1.4 \\ 4.9 & -0.4 & -1.9 & -8.9 & 6.6 & 2.6 & 3.6 & 3.6 \\ 0.7 & -0.9 & -0.9 & 4.1 & 7.2 & -15.5 & 2.8 & 1.8 \\ -3.1 & -1.0 & -2.5 & -5.5 & -5.2 & -6.7 & 10.7 & -1.1 \\ -2.9 & -0.2 & -1.2 & -2.7 & 3.6 & 2.6 & 0.4 & -6.4 \end{pmatrix}$$

- Quantize (round($D./Q$))

$$Dq = \begin{pmatrix} 6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 1 & -2 & 0 & 0 & 0 & 0 & 0 \\ 1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- zigzag ordering: 6 1 4 1 1 0 -2 -2 allzeros

Example Huffman Table for Lena

Coding the Example Block

- zigzag ordering: 6 1 4 1 1 0 0 -2 -2 allzeros
- Huffman symbols: [run, category]
DC (0,1) (0,3) (0,1) (0,1) (2,2) (0,2) EOB
- Code:
(DC diff. code)/001/100100/001/001/111110010/0100/1010
- Bitrate (assuming 8 bits for the DC residual symbol):
 $\frac{40}{64} = 0.625$ bits/pixel

Decoding

1. Entropy-decode the bitstream back to the quantized blocks
2. Dequantize: multiply each block coefficient by the corresponding coefficient of the quantization matrix
3. Apply the inverse DCT transform on each block
4. Denormalize: add 128 to each coefficient

- Example: Reconstructed block \hat{B}

$$\begin{pmatrix} 143 & 147 & 153 & 157 & 157 & 154 & 149 & 145 \\ 145 & 147 & 151 & 154 & 153 & 149 & 144 & 141 \\ 146 & 147 & 149 & 149 & 146 & 142 & 137 & 134 \\ 146 & 146 & 145 & 142 & 139 & 135 & 132 & 130 \\ 145 & 143 & 140 & 136 & 133 & 131 & 130 & 130 \\ 141 & 138 & 134 & 131 & 130 & 131 & 134 & 135 \\ 136 & 134 & 130 & 128 & 129 & 133 & 139 & 142 \\ 133 & 131 & 127 & 126 & 129 & 135 & 142 & 147 \end{pmatrix}$$

- Error block

$$\begin{pmatrix} 0 & 0 & -4 & -5 & -1 & -7 & -3 & 4 \\ 6 & -1 & -8 & 0 & -5 & -5 & 9 & -9 \\ 1 & -4 & -4 & 0 & -2 & 3 & -9 & -1 \\ 6 & -1 & 0 & 2 & 7 & -1 & -2 & 7 \\ 1 & 0 & 2 & 11 & -9 & -4 & 9 & 8 \\ -2 & 7 & 5 & -4 & -4 & 4 & 5 & 6 \\ 9 & 3 & -6 & 2 & 9 & 3 & 1 & 2 \\ 11 & -7 & 9 & 8 & 8 & 4 & 0 & -2 \end{pmatrix}$$

- MSE=5.2

Extended JPEG

- Extended JPEG allows for several optional enhancements:
 - 12-bit/pixel input
 - Arithmetic coding is allowed as an alternative to Huffman coding
 - Adaptive quantization: Allows 5-bit scale change to the quantization matrix from one block to another
 - SPIFF: A file format that provides for the interchange of compressed image files between different application environments
 - Progressive transmission (PT)
 - * Sequential PT
 - Spectral selection
 - Successive approximations
 - * Hierarchical PT (pyramid encoding)
 - Tiling
 - Selective refinement

Sequential Progressive Transmission

Hierarchical Progressive Transmission

- Subsampling filters are left to the users to specify
- Upsampling: bilinear interpolation

$$\begin{array}{cc} A & B \\ C & D \end{array} \longrightarrow \begin{array}{ccc} A & a & B \\ c & x & b \\ C & d & D \end{array}$$

$$a = \frac{A+B}{2}, b = \frac{B+D}{2}, c = \frac{A+C}{2}, d = \frac{C+D}{2}, x = \frac{A+B+C+D}{4}$$

MPEG

(History)

Standard Activity	Description
H.261 Dec. 1990	Internation Telecommunication Union (ITU); designed for ISDN applications at $p \times 64$ kbits/sec, ($p = 1, 2, \dots, 30$)
H.263 Feb. 1995	ITU Recommendation; video coding for low bitrate communication; improved visual telephony applications
MPEG Nov. 1994	ISO Moving Picture Expert Group; MPEG1: storage/retrieval of video+audio at about 1.5 Mbits/s MPEG2: for generic applications at higher birates (initially 10Mbits/s); MPEG3: initially intended for 50 Mbits/s bitrate, but MPEG2 was found to achieve the MPEG3 speed goals, leading to the abandonment of MPEG3

Basic Concepts

- Video \equiv a sequence of images called frames
- Color
 - Three components: red (R), green (G), and blue (B)
 - For compatibility with non-colored media, the RGB model was converted to an equivalent model — YC_bC_r
 - * Y is the luminance component, which was experimentally determined to be

$$Y = 0.299R + 0.587G + 0.114B$$

$$* C_b = B - Y$$

$$* C_r = R - Y$$

- Y is referred to as *luma*, and C_b & C_r as *chroma*
- Every frame is really 3 images: one Y , one C_b and one C_r
- 8 bits/pixel for each of the three color components
- Because human vision is less sensitive to color, the C_b and C_r images are downsampled by 2 in each dimension (they are quarter the size of Y images)
- Much of MPEG processing is on the basis of a *macroblock*: A 16×16 luminance block with the two 8×8 associated chroma blocks

Modes of MPEG Compression

- Intraframe compression
 - Exploits spatial redundancy only
 - Operates on single frames independently of other frames
- Interframe compression
 - Exploits both spatial and temporal redundancies
 - Employs motion estimation (MS) without standardizing any MS algorithm
 - Derives motion-compensated predictions of frames
 - Finally, it performs JPEG-like compression on the residual frames

Types of Frames in MPEG

- MPEG has 3 types of frames: I, P, and B
- I frames are strictly intra compressed as in JPEG. Their purpose is to provide random access points to the video
- P frames are motion-compensated forward-predictive-coded frames; they are interframe compressed, and typically provide more compression than I frames
- B frames are motion-compensated bidirectionally-predictive-coded frames; they are interframe compressed, and typically provide the most compression
- The relative numbers of I, P and B frames are arbitrary
- An I frame must occur at least once every 132 frames to provide user-acceptable speed of random access to various parts of a video

Interframe Compression of P and B Blocks

- A motion-compensated prediction (i.e., approximation) \overline{f} of a P/B frame f is made
- The residual $f - \overline{f}$ is then compressed in a JPEG-like style
- Any macroblock of the original frame f may be strictly intra compressed if its prediction is deemed to be poor
- Issues to be addressed
 - Method of strict intra compression
 - Method of compressing residual macroblocks
 - Motion-compensated prediction

Flowchart of MPEG Compression

Motion-Estimation and Prediction

- Motion estimation is performed on the basis of macroblocks, using the 16×16 luminance blocks only
- Motion is assumed to be uniform across all the pixels of a macroblock
- Remark: There is a tradeoff in deciding the size of the basic block for motion estimation
 - The block has to be sufficiently large to avoid “false hits”
 - The block has to be sufficiently small to avoid diverse motions within one single block
 - The MPEG block size, 16×16 , is a good compromise

Motion-Estimation and Prediction for P Frames

- Consider a P-frame P
- P will be predicted (i.e., approximated) from one single reference frame R
- R is the most recent (decoded) I or P frame
- For each macroblock MB of P , find the closest matching macroblock MB' in the reference frame R
- If the MB-to-MB' match is satisfactory, then
 - treat MB' as the prediction (i.e., approximation) of MB
 - record the motion (i.e., displacement) vector between the two macroblocks (allowing half-pixel accuracy)
 - Compute and compress the macroblock residual MB–MB' (luma and chroma)

- If the MB-to-MB' match is found to be unsatisfactory, then the macroblock MB is strictly intra compressed as is done in I frames
- The motion vectors of all the macroblocks of P exhibit redundancy due to similar (or sometimes identical) motion experienced by many neighboring macroblocks
- This redundancy is exploited by coding the consecutive differential values of motion vectors (i.e., DPCM)
- Remark 1: MPEG does not standardize the decision mechanism for judging whether or not a match between two macroblocks is satisfactory
- Remark 2: A typical decision mechanism involves computing an error measure between the luminance of the two macro blocks. The match is treated as satisfactory if and only if the error is below a certain threshold. Possible error measures include mean-square error (MSE), mean absolute-difference error (MAD), and $\frac{\text{variance}(MB-MB')}{\text{variance}(MB)}$.

Motion-Estimation and Prediction for B Frames

- Consider a B-frame B
- B will be predicted (i.e., approximated) from TWO reference frames R_1 and R_2
- R_1 is the most recent (decoded) past I/P frame, and R_2 is the nearest (decoded) future I/P frame
- For each macroblock MB of B , find the closest matching macroblock MB_1 in the reference frame R_1 , and the closest matching macroblock MB_2 in R_2
- The predicted macroblock is $PM = NINT(\alpha_1 MB_1 + \alpha_2 MB_2)$
 - $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$ if both matches are satisfactory
 - $\alpha_1 = 1$ and $\alpha_2 = 0$ if only the 1st match is satisfactory
 - $\alpha_1 = 0$ and $\alpha_2 = 1$ if only the 2nd match is satisfactory
 - $\alpha_1 = 0$ and $\alpha_2 = 0$ if neither match is satisfactory

- Compute and compress the macroblock residual $MB - PM$ (luma and chroma)
- If neither match is found to be satisfactory, then the macroblock MB is strictly intra compressed as is done in I frames
- Record the motion vector(s) between the MB and the other one or two macroblocks (allowing half-pixel accuracy)
- Again, the motion-vector redundancy is exploited by coding the consecutive differential values of motion vectors (i.e., DPCM)
- Remark: The prediction mode chosen is 2-bit coded and passed on along with the macroblock header information

MPEG1 Constrained Parameters

horizontal picture size	≤ 768
vertical picture size	≤ 576
picture area	≤ 396 macroblocks at 25 frames/sec
picture area	≤ 330 macroblocks at 30 frames/sec
picture rate	≤ 30 frames/sec
bitrate	≤ 1.856 Mbits/sec (constant)

MPEG2

- Higher data rates than MPEG1
- MPEG2 allows for higher quality source images
 - 4:2:0 (chroma subsamples only horizontally)
 - 4:4:4 (no subsampling of chroma)
 - Note: 4:2:2 is what's supported by MPEG1
- MPEG2 allows for finer quantization and for specifying separate quantization table for luma and chroma
- MPEG2 allows for finer adjustment of quantization scale factor, used in intra compression
- MPEG2 allows for interlaced video
- MPEG2 supports error concealment (of lost macroblocks)
- MPEG2 supports scalable compression
 - SNR-scalability: by sending bands of DCT coefficients
 - spatial-scalability: pixel resolution by down-/up-sampling
 - temporal-scalability: different frame rates by skipping frames

MPEG2 (Cont.)

- MPEG2 has a *Profile* and *Level* structure
 - Profiles are algorithmic elements included in MPEG2
 - Levels are upper bounds on parameter values
- Profiles (profiles are backward-compatible)
 1. Simple: No use of B frames
 2. Main: what was described earlier, but no scalability
 3. SNR-scalable
 4. Spatially scalable
 5. High: temporally scalable, higher-quality source data (4:2:0, 4:2:2 and possibly 4:4:4 in the future)
- Levels
 1. Low: 352×240 frame size
 2. Main: 720×480 frame size
 3. High 1440: 1440×1152 frame size
 4. High: 1920×1080 frame size

Allowed Level-Profile Combinations

	simple	main	SNR- scalable	spatially scalable	high
high	NO	YES	No	No	YES
high 1440	NO	YES	No	YES	YES
main	YES	YES	YES	NO	YES
low	NO	YES	YES	NO	NO

- The Grand Alliance (for HDTV) follows the Main Profile and High Level combination, for a maximum bitrate of 45 Mbits/sec

References

1. B. pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, Van Nostrand reinhold, New York 1993.
2. ISO-11172-2: Generic Coding of moving pictures and associated audio (MPEG-1)
3. ISO-13818-2: Generic Coding of moving pictures and associated audio (MPEG-2)